Πανεπιστήμιο Θεσσαλίας
Τμήμα Πληροφορικής

Αρχιτεκτονική Υπολογιστών

Ασκήσεις Χειμερινού Εξαμήνου 2018-2019
(μέρος Β΄)

Άσκηση 6:

Στην άσκηση αυτή θα μελετήσουμε την εκτέλεση ενός βρόχου διανυσματικής επεξεργασίας, που αποτιμά τη έκφραση W = a × (b × X - c × Y) / Z + d × U για αριθμούς διπλής ακρίβειας. Θεωρήστε τον παρακάτω κώδικα MIPS που υλοποιεί αυτόν το βρόχο:

```
foo:  ldc1   $f3,0($1)
      mult.d $f4,$f1,$f3
      ldc1   $f5,0($2)
      mult.d $f6,$f2,$f5
      sub.d  $f7,$f4,$f6
      mult.d $f4,$f0,$f7
      ldc1   $f3,0($3)
      bc1z   $f3,exfoo
      div.d  $f7,$f4,$f3
      ldc1   $f4,0($4)
      mult.d $f5,$f8,$f4
      add.d  $f6,$f7,$f5
      sdc1   $f6,0($5)
      daddiu $5,$5,8
      daddiu $4,$4,8
      daddiu $3,$3,8
      daddiu $2,$2,8
      daddiu $1,$1,8
      bne    $1,$6,foo
```

όπου η bc1z είναι εντολή διακλάδωσης με συνθήκη τη μηδενική τιμή κάποιου καταχωρητή κινητής υποδιαστολής και exfoo είναι κάποια διεύθυνση κώδικα έξω από το βρόχο. Στην αρχή της εκτέλεσης του βρόχου οι καταχωρητές $f0, $f1, $f2 και $f8 περιέχουν τους αριθμούς a, b, c και d αντίστοιχα, οι καταχωρητές $1, $2, $3, $4 και $5 περιέχουν τις αρχικές διευθύνσεις των διανυσμάτων X, Y, Z, U και W αντίστοιχα, και ο καταχωρητής $6 περιέχει κατάλληλη τιμή για τερματισμό του βρόχου.

Υποθέστε ότι ο κώδικας αυτός εκτελείται σε έναν επεξεργαστή MIPS, όπου:

- Όλες οι εντολές σταθερής υποδιαστολής έχουν φάση εκτέλεσης διάρκειας 1 κύκλου μηχανής. Η φάση προσπέλασης μνήμης εντολών φόρτωσης και αποθήκευσης απαιτεί 1 ακόμα κύκλο μηχανής στη φάση εκτέλεσης αυτών των εντολών.
- Οι εντολές διακλάδωσης εκτελούνται στη φάση εκτέλεσης, χωρίς κύκλο καθυστέρησης, με επιτυχημένη πρόβλεψη *μη* εκτέλεσης άλματος για την bc1z και εκτέλεσης άλματος για την bne. Χωρίς υποθετική εκτέλεση, εντολές που ακολουθούν μια διακλάδωση δε μπαίνουν στη φάση εκτέλεσης πριν την εκτέλεση της διακλάδωσης.
- Οι εντολές κινητής υποδιαστολής εκτελούνται σε μια μονάδα πρόσθεσης, μια μονάδα πολλαπλασιασμού και μια μονάδα διαίρεσης, σε χρόνους 2, 6 και 12 κύκλων μηχανής αντίστοιχα. Οι δύο πρώτες μονάδες υποστηρίζουν μερική επικάλυψη διαδοχικών πράξεων, επιτρέποντας νέες πράξεις να ξεκινάνε σε κάθε κύκλο μηχανής για την πρώτη, και κάθε 3 κύκλους μηχανής για τη δεύτερη. Η μονάδα διαίρεσης δεν υποστηρίζει επικάλυψη.
- Η μνήμη δεν επιτρέπει πάνω από μία προσπέλαση ανά κύκλο μηχανής, με τις εντολές φόρτωσης να έχουν μεγαλύτερη προτεραιότητα από τις εντολές αποθήκευσης. Υποθέστε ότι

οι περιοχές μνήμης όπου βρίσκονται τα πέντε διανύσματα δεν επικαλύπτονται σε καμία επανάληψη του βρόχου.
- Σε κάθε περίπτωση κινδύνου δομικής εξάρτησης οι εντολές εξυπηρετούνται σε σειρά FIFO.

Α. Έστω ότι ο επεξεργαστής εκτελεί εντολές εκτός σειράς με τη βοήθεια πίνακα παρακολούθησης (scoreboard), με τη γενική οργάνωση που δίνεται στο βιβλίο, με τη διαφορά ότι ο αριθμός μονάδων κινητής υποδιαστολής είναι αυτός που αναφέρθηκε πιο πάνω, ενώ οι μονάδες σταθερής υποδιαστολής είναι τρεις. Δείξτε την εκτέλεση των τριών πρώτων επαναλήψεων του βρόχου, συμπληρώνοντας σε έναν πίνακα για κάθε διαδοχική εντολή που εκτελείται:
- τον αριθμό επανάληψης του βρόχου,
- τον κύκλο μηχανής της έκδοσης στον πίνακα παρακολούθησης,
- τον κύκλο μηχανής της ανάγνωσης τελούμενων,
- τον κύκλο μηχανής της ολοκλήρωσης της φάσης εκτέλεσης,
- τον κύκλο μηχανής της αποθήκευσης αποτελέσματος στο φάκελο καταχωρητών, και
- την αιτία καθυστέρησης – αν υπάρχει – μεταξύ των κύκλων αυτών,

όταν η έκδοση, η ανάγνωση τελούμενων και η αποθήκευση αποτελέσματος απαιτούν καθεμία από 1 κύκλο μηχανής. Δώστε το περιεχόμενο του πίνακα παρακολούθησης κάθε στιγμή που η εντολή sdc1 βρίσκεται στη φάση αποθήκευσης αποτελέσματος. Πόσους κύκλους μηχανής εκτιμάτε ότι χρειάζεται κατά μέσο όρο κάθε επανάληψη του βρόχου;

Β. Να επαναλάβετε το προηγούμενο ερώτημα, μετά από μετονομασία καταχωρητών για τις εντολές που εμφανίζουν κινδύνους τύπου ΕΜΕ, οι οποίοι καθυστερούν την έκδοση αυτών στον πίνακα παρακολούθησης. Πόσο επιταχύνεται ο κώδικας εξαιτίας της μετονομασίας; Πώς θα μπορούσε να επιταχυνθεί περισσότερο;

Γ. Θεωρήστε τώρα ότι ο επεξεργαστής εκτελεί εντολές εκτός σειράς με τη βοήθεια σταθμών δέσμευσης και δρομολόγηση με τον αλγόριθμο του Tomasulo χωρίς υποθετική εκτέλεση, και με τη γενική οργάνωση και αριθμό σταθμών δέσμευσης που δίνεται στο βιβλίο, αλλά με αριθμό μονάδων κινητής υποδιαστολής αυτόν που αναφέρθηκε πιο πάνω, και με την προσθήκη της μονάδας διαίρεσης με δύο σταθμούς δέσμευσης. Υποθέστε ότι όλες οι εντολές σταθερής υποδιαστολής εκτός των εντολών προσπέλασης μνήμης εκτελούνται σε τρεις ανεξάρτητες υπομονάδες, με ένα σταθμό δέσμευσης για κάθε μονάδα. Συμπληρώστε σε έναν πίνακα για καθεμία από τις εντολές των τριών πρώτων επαναλήψεων του βρόχου που εκτελείται:
- τον αριθμό επανάληψης του βρόχου,
- τον κύκλο μηχανής της έκδοσης στον αντίστοιχο σταθμό δέσμευσης,
- τον κύκλο μηχανής της φάσης εκτέλεσης,
- τον κύκλο μηχανής της προσπέλασης μνήμης – αν υπάρχει,
- τον κύκλο μηχανής της χρήσης του διαύλου CDB – αν υπάρχει, και
- την αιτία καθυστέρησης – αν υπάρχει – μεταξύ των κύκλων αυτών,

όταν η έκδοση και η χρήση του CDB απαιτούν καθεμία από 1 κύκλο μηχανής. Δώστε το περιεχόμενο των σταθμών δέσμευσης κάθε στιγμή που η εντολή sdc1 βρίσκεται στη φάση αποθήκευσης αποτελέσματος. Πόσους κύκλους μηχανής εκτιμάτε ότι χρειάζεται κατά μέσο όρο κάθε επανάληψη του βρόχου;

Δ. Να επαναλάβετε το προηγούμενο ερώτημα, για επεξεργαστή που υποστηρίζει υποθετική εκτέλεση με τη γενική οργάνωση και αριθμό σταθμών δέσμευσης που δίνεται στο βιβλίο, με τις τροποποιήσεις που αναφέρθηκαν παραπάνω, δίνοντας και τον κύκλο μηχανής απόσυρσης κάθε εντολής. Πόσο επιταχύνεται ο κώδικας με την υποθετική εκτέλεση;

Ε. Να επαναλάβετε το ερώτημα Δ, επεκτείνοντας τον επεξεργαστή σε υπερβαθμωτό βαθμού 2, υποθέτοντας ότι έχει 2 διαύλους CDB, και ότι ο πίνακας επαναδιάταξης μπορεί να αποσύρει μέχρι 2 εντολές σε κάθε κύκλο μηχανής. Πόσο επιταχύνεται ο κώδικας σε σχέση με το προηγούμενο ερώτημα; Είναι η επιτάχυνση κοντά στο βαθμό του επεξεργαστή, και γιατί;

Άσκηση 7:

Α. Θεωρήστε τον παρακάτω κώδικα C:

```
i = k = 0;
while (x = a[i++]) {
    if ((x!='a')||(x=='b')) k=k+1;
    else if ((x>'j')&&(x<'n')&&(x<'q')) k=k+2;
    else if (x=='z') k=-1;
    else if (x==' ') continue;
    if (k>=0) continue;
    printf("done\n");
    break;
}
```

ο οποίος επεξεργάζεται μια συμβολοσειρά τοποθετημένη στον πίνακα a στοιχείων τύπου char. Η μεταβλητή x είναι τύπου char, ενώ οι μεταβλητές i και k είναι ακέραιες.
Ο κώδικας μεταφράζεται για επεξεργαστή MIPS. Ο πίνακας a τοποθετείται στη μνήμη σε μετατόπιση 800 σχετικά με τον $gp, η μεταβλητή i τοποθετείται στον καταχωρητή $12, η μεταβλητή k τοποθετείται στον καταχωρητή $15 και η μεταβλητή x τοποθετείται στον καταχωρητή $8. Το όνομα "printf" αντιστοιχεί στη συνάρτηση βιβλιοθήκης "_printf" που καλείται με υποχρεωτική πρώτη παράμετρο τη διεύθυνση μιας συμβολοσειράς, και προαιρετικές άλλες παραμέτρους που εδώ δεν υπάρχουν. Η σταθερά "done\n" είναι τοποθετημένη στη μνήμη σε μετατόπιση 124 σχετικά με τον $gp.
Γράψτε τον κώδικα MIPS που προκύπτει από τον παραπάνω κώδικα C χωρίς καμία βελτιστοποίηση, παρά μόνο με τη βραχυκύκλωση των λογικών εκφράσεων που υποστηρίζει η C. Στη συνέχεια βρείτε όλες τις συσχετίσεις που υπάρχουν στην εκτέλεση των εντολών διακλάδωσης που θα έχουν παραχθεί. Αν ο κώδικας πρόκειται να εκτελεστεί σε έναν επεξεργαστή με δυναμική πρόβλεψη διακλαδώσεων δύο επιπέδων με δύο ψηφία τοπικής ιστορίας ανά διακλάδωση και ανά τιμή καθολικής ιστορίας, πόσα ψηφία καθολική ιστορίας είναι αρκετά για πλήρη εκμετάλλευση των συσχετίσεων στις εντολές διακλάδωσης του κώδικα; Υποθέτοντας αρκετά μεγάλο πίνακα ιστορίας, ώστε να μην υπάρξουν συγκρούσεις στις διευθύνσεις διακλάδωσης, δείξτε το περιεχόμενο των θέσεων του πίνακα που επηρεάζεται από τη συσχέτιση μετά από ικανό αριθμό επαναλήψεων, ώστε η συσχέτιση να έχει πλήρως εκδηλωθεί.

Β. Να επαναλάβετε το προηγούμενο ερώτημα, αν στον κώδικα C που σας δόθηκε δεν υπάρχουν οι λέξεις-κλειδιά "else".

Άσκηση 8 (εργαστηριακή):

Γράψτε ένα πρόγραμμα προσομοίωσης για την αξιολόγηση της δυναμικής δρομολόγησης εντολών με τον αλγόριθμο του Tomasulo χωρίς υποθετική εκτέλεση πάνω σε έναν επεξεργαστή MIPS, όπως αυτόν που περιγράφεται στο βιβλίο. Αγνοήστε την προσκόμιση και αποκωδικοποίηση των εντολών, θεωρώντας ότι οι εντολές βρίσκονται αρχικά όλες στο παράθυρο εντολών. Υποθέστε ότι η ΜΕΔ του επεξεργαστή διαθέτει από μία υπομονάδα εκτέλεσης για κάθε λειτουργία του παρακάτω πίνακα, ο οποίος δίνει και τον αριθμό σταθμών δέσμευσης και το χρόνο εκτέλεσης αυτής σε κύκλους μηχανής:

Υπομονάδα	Σταθμοί δέσμευσης	Χρόνος εκτέλεσης
Integer	3	2
Branch	2	4
Load-store	6	4
FP add	3	6
FP multiply	3	12
FP divide	3	24

Θεωρήστε ότι όλες οι υπομονάδες εκτέλεσης – εκτός από τις Load-store, FP multiply και FP divide – είναι μερικά επικαλυπτόμενες με ιδανικό ρυθμό εκτέλεσης πράξεων, ώστε κάθε εντολή να μπορεί να αρχίσει την εκτέλεσή της, αμέσως μόλις τα δεδομένα που χρειάζεται γίνουν διαθέσιμα. Ειδικά για τις τρεις πιο πάνω υπομονάδες, η υλοποίησή τους επιτρέπει επικάλυψη με έναρξη εκτέλεσης μιας νέας όμοιας λειτουργίας στο δεύτερο για την πρώτη, στον τρίτο και στον έβδομο για τη δεύτερη, και στο εντέκατο κύκλο μηχανής για την τρίτη, μετά την έναρξη της προηγούμενης λειτουργίας. Οι εντολές προσπέλασης μνήμης εκτελούνται σε πρώτη φάση ως ακέραιες για τον υπολογισμό διεύθυνσης, και στη συνέχεια η εκτέλεσή τους συνεχίζεται στην υπομονάδα Load-store. Θεωρήστε ακόμα ότι οι εντολές διακλάδωσης έχουν πάντα επιτυχημένη πρόβλεψη.

Για την προσομοίωση, αγνοήστε τις τιμές των δεδομένων, ώστε να επικεντρώσετε την ανάλυσή σας στη δρομολόγηση και εκτέλεση των εντολών. Όμως, για ανίχνευση εξαρτήσεων μνήμης, υποθέστε ότι η διεύθυνση λαμβάνει τυχαία τιμή από 0 μέχρι N-1, για κάποια τιμή του N, η οποία θα γράφεται στο πεδίο A των σταθμών δέσμευσης της υπομονάδας Load-store. Δώστε προτεραιότητα στις εντολές φόρτωσης, ώστε αυτές να πηγαίνουν πρώτες στη μνήμη όποτε τίθεται τέτοιο θέμα. Θα πρέπει όμως να ελέγχετε αν η διεύθυνση χρησιμοποιείται από προηγούμενη εντολή αποθήκευσης που δεν έχει ακόμα ξεκινήσει εγγραφή στη μνήμη. Σε τέτοια περίπτωση, η ανάγνωση θα θεωρείται ότι γίνεται με παροχέτευση από το σταθμό της εντολής αποθήκευσης και θα ολοκληρώνεται σε έναν κύκλο μηχανής από τη στιγμή που το δεδομένο αποθήκευσης γίνεται διαθέσιμο. Σε περίπτωση που μια προηγούμενη εντολή αποθήκευσης δεν έχει ακόμα εκτελέσει την πρώτη φάση εκτέλεσής της, καμία επόμενη εντολή προσπέλασης μνήμης δε μπορεί να προχωρήσει στη δεύτερη φάση εκτέλεσής της. Η επιλογή μεταξύ εντολών φόρτωσης που είναι έτοιμες για προσπέλαση μνήμης γίνεται με σειρά FIFO σε σχέση με την έκδοσή τους, το ίδιο και για εντολές αποθήκευσης.

Για να κωδικοποιήσετε τον αλγόριθμο του Tomasulo, χρησιμοποιήστε το σχετικό αλγόριθμο του βιβλίου. Θεωρήστε ότι ο φάκελος καταχωρητών έχει απεριόριστο αριθμό θυρών ανάγνωσης και εγγραφής.
Για την αξιολόγηση της επίδοσης του αλγόριθμου Tomasulo, το πρόγραμμά σας θα πρέπει να μπορεί να προσομοιώνει και στατική δρομολόγηση εντολών.

Η είσοδος του προγράμματός σας θα είναι οι 4 μικροί βρόχοι που βρίσκονται στα αρχεία loop1.txt, loop2.txt, loop3.txt και loop4.txt. Για απλούστευση, μπορείτε αν θέλετε να τροποποιήσετε τον κώδικα με το χέρι, ώστε αντί εντολών συμβολικής γλώσσας, το πρόγραμμά σας να διαβάζει για κάθε εντολή έναν κώδικα υπομονάδας και τους αριθμούς καταχωρητών που αποτελούν τα τελούμενα της εντολής. Οι καταχωρητές κινητής υποδιαστολής μπορούν απλά να κωδικοποιηθούν με αριθμούς 32-63. Θεωρήστε ότι οι εντολές διακλάδωσης που βρίσκονται στο σώμα των βρόχων δεν εκτελούν ποτέ άλμα, ότι οι εντολές διακλάδωσης στο τέλος των βρόχων εκτελούν πάντα άλμα, και ότι κάθε βρόχος εκτελεί 1000 επαναλήψεις.

Α. Εκτελέστε το πρόγραμμα που γράψατε πάνω στους 4 βρόχους, πρώτα με στατική και μετά με δυναμική δρομολόγηση. Σχολιάστε τις διαφορές στις δύο τεχνικές δρομολόγησης. Δοκιμάστε τουλάχιστον δύο τιμές για το N, μία πολύ μικρή και μία πολύ μεγάλη, ώστε να δείτε πώς επηρεάζεται η δρομολόγηση από τις εξαρτήσεις μνήμης.

Β. Στη συνέχεια, προσπαθήστε να εφαρμόσετε με το χέρι τεχνικές μετασχηματισμού στους κώδικες των 4 βρόχων, όπως για παράδειγμα πραγματικό και συμβολικό ξεδίπλωμα, ώστε να βελτιώσετε τη συμπεριφορά της στατικής δρομολόγησης όσο μπορείτε περισσότερο, και ξαναεκτελέστε το πρόγραμμα. Σχολιάστε τις διαφορές που παρατηρείτε σε σχέση με την προηγούμενη εκτέλεση.

Άσκηση 9:

Θεωρήστε την τεχνική του ξεδιπλώματος βρόχων για βελτίωση στη δρομολόγηση των εντολών τους, όταν αυτή *δε* γίνεται δυναμικά από το υλικό. Για μεταβλητό αριθμό επαναλήψεων N, και

για τυχαίο βαθμό ξεδιπλώματος k, η τεχνική αυτή εφαρμόζεται με έναν από τους ακόλουθους δύο τρόπους:
(α) με υλοποίηση δύο βρόχων, όπου ο ένας είναι ο ξεδιπλωμένος που εκτελεί N/k επαναλήψεις, και ο άλλος είναι μη ξεδιπλωμένος και εκτελεί τις υπόλοιπες επαναλήψεις του αρχικού βρόχου, και
(β) με υλοποίηση ενός μόνο ξεδιπλωμένου βρόχου που εκτελεί $\lceil N/k \rceil$ επαναλήψεις.
Καθένας από τους δύο παραπάνω τρόπους αναδιατάσσει τις εντολές του σώματος του βρόχου που προκύπτει για βέλτιστη δρομολόγηση, και υλοποιεί τον ελάχιστο δυνατό αριθμό εντολών ελέγχου του βρόχου.

Χρησιμοποιώντας τον ακόλουθο βρόχο MIPS:

```
loop: ldc1  $f0,0($1)
      ldc1  $f1,0($2)
      add.d $f2,$f0,$f1
      sub.d $f3,$f2,$f4
      sdc1  $f3,0($1)
      addiu $1,$1,-8
      addiu $2,$2,-8
      bne   $1,$0,loop
```

εφαρμόστε την τεχνική του ξεδιπλώματος με τους δύο πιο πάνω τρόπους για k = 4, και δείξτε σε τι διαφέρουν οι δύο ξεδιπλωμένοι βρόχοι, ώστε να μπορούν να εκτελούν συνολικά ακριβώς N επαναλήψεις του αρχικού βρόχου. Σχολιάστε τη διαφορά στην απόδοση των δύο τρόπων ξεδιπλώματος, και αναφέρετε τι υποστήριξη θα χρειαστείτε από το υλικό, ώστε οι δύο αυτοί τρόποι να έχουν παραπλήσια απόδοση.

Ποιος τρόπος αναμένετε να απαιτεί μεγαλύτερο βαθμό ξεδιπλώματος για βέλτιστη δρομολόγηση εντολών και γιατί; Επαληθεύσατε την απάντησή σας για την περίπτωση που ο παραπάνω κώδικας εκτελείται σε ένα βαθμωτό επεξεργαστή MIPS, όπου εκτός της υπομονάδας σταθερής υποδιαστολής, μια υπομονάδα κινητής υποδιαστολής εκτελεί πράξεις σε 4 κύκλους μηχανής με ιδανική επικάλυψη, ώστε μια νέα πράξη να μπορεί να ξεκινά σε κάθε κύκλο μηχανής.

Άσκηση 10:

Μελετήστε τις δύο δημοσιεύσεις από το IBM Journal του 1967 που περιγράφουν τον υπολογιστή IBM System/360, δίνοντας ιδιαίτερη προσοχή στη δυναμική δρομολόγηση εντολών για εκτέλεση εκτός σειράς με τον αυθεντικό αλγόριθμο του Tomasulo. Στη συνέχεια επιλέξτε κάποιον σχετικά σύγχρονο υπερβαθμωτό επεξεργαστή με δυναμική δρομολόγηση εντολών (για παράδειγμα MIPS R10K, PowerPC G4, Ultra Sparc III, Alpha 21364, Intel Core i7), και αναζητήστε στο Διαδίκτυο κάποια δημοσίευση ή εγχειρίδιο που να περιγράφει την αρχιτεκτονική του και ειδικότερα την τεχνική δρομολόγησης εντολών που χρησιμοποιεί. Συγκρίνετε τις δύο αρχιτεκτονικές, δίνοντας τα βασικά χαρακτηριστικά τους, και σχολιάστε την εξέλιξη στη δυναμική δρομολόγηση εντολών από την αρχιτεκτονική του 1967 στη σημερινή.